IN THE CLAIMS

Please amend the claims as follows.

1. (Previously Presented) A data processor having a clustered architecture comprising:

a branching cluster and a non-branching cluster, each capable of executing instructions and computing branch conditions, said branching cluster operable to perform branch address computations for said branching cluster and said non-branching cluster, the non-branching cluster incapable of performing branch address computations; and

remote conditional branching control circuitry that causes said branching cluster to perform a branch address computation in response to sensing a conditional branch instruction in said non-branching cluster, and that communicates a computed branch condition from said non-branching cluster to said branching cluster.

2. (Original) The data processor as set forth in Claim 1 wherein each of said branching cluster and said non-branching cluster comprises at least one register file.

3. (Original) The data processor as set forth in Claim 1 wherein each of said branching cluster and said non-branching cluster comprises an instruction execution pipeline comprising N processing stages, each of said N processing stages capable of performing at least one of a plurality of execution steps associated with a pending instruction being executed by said instruction execution pipeline.

4. (Original) The data processor as set forth in Claim 1 wherein said remote conditional branching control circuitry further causes said branching cluster to perform a next program counter address computation in response to sensing a conditional branch instruction in said non-branching cluster.

5. (Original) The data processor as set forth in Claim 4 wherein said remote conditional branching control circuitry selects one of said computed next program counter address and said computed branch address in response to said computed branch condition.

6. (Original) The data processor as set forth in Claim 5 wherein said remote conditional branching control circuitry comprises a multiplexor that is responsive to said computed branch condition.

7. (Original) The data processor as set forth in Claim 1 wherein said data processor issues a shadow conditional branch instruction in said branching cluster to perform said branch address computation in response to sensing said conditional branch instruction in said non-branching cluster.

8. (Previously Presented) For use in a data processor comprising a branching cluster and a non-branching cluster, each capable of executing instructions and computing branch conditions, said branching cluster operable to perform branch address computations for said branching cluster and said non-branching cluster, a method of operating said data processor comprising the steps of:

computing a branch address in the branching cluster in response to sensing a conditional branch instruction in said non-branching cluster, the non-branching cluster incapable of performing branch address computations; and

communicating a branch condition computed by said non-branching cluster from said non-branching cluster to said branching cluster.

9. (Original) The method of operating said data processor as set forth in Claim 8 further comprising the step of computing said branch condition in said non-branching cluster.

10. (Original) The method of operating said data processor as set forth in Claim 9 further comprising the step of computing a next program counter address.

11. (Original) The method of operating said data processor as set forth in Claim 10 further comprising the step of selecting one of said computed next program counter address and said computed branch address in response to said computed branch condition.

12. (Original) The method of operating said data processor as set forth in Claim 8 wherein each of said branching cluster and said non-branching cluster comprises an instruction execution pipeline comprising N processing stages, said method further comprising the step of performing in each of said N processing stages at least one of a plurality of execution steps associated with a pending instruction being executed by said instruction execution pipeline.

13. (Original) The method of operating said data processor as set forth in Claim 8 further comprising the step of issuing a shadow conditional branch instruction in said branching cluster to perform said branch address computation in response to sensing said conditional branch instruction in said non-branching cluster.

14. (Previously Presented) A processing system comprising:

a data processor having a clustered architecture;

a memory associated with said data processor;

a plurality of peripheral circuits associated with said data processor for performing selected functions in association with said data processor;

wherein said data processor comprises:

at least a branching cluster and a non-branching cluster that are each capable of executing instructions and computing branch conditions, said branching cluster operable to perform branch address computations for said at least said branching cluster and said non-branching cluster, the non-branching cluster incapable of performing branch address computations; and

remote conditional branching control circuitry that causes said branching cluster to perform a branch address computation in response to sensing a conditional branch instruction in said non-branching cluster, and that communicates a computed branch condition from said non-branching cluster to said branching cluster.

15. (Original) The processing system as set forth in Claim 14 wherein each of said branching cluster and said non-branching cluster comprises at least one register file.

16. (Original) The processing system as set forth in Claim 14 wherein each of said at least said branching cluster and said non-branching cluster comprises an instruction execution pipeline comprising N processing stages, each of said N processing stages capable of performing at least one of a plurality of execution steps associated with a pending instruction being executed by said instruction execution pipeline.

17. (Original) The processing system as set forth in Claim 14 wherein said remote conditional branching control circuitry further causes said branching cluster to perform a next program counter address computation in response to sensing a conditional branch instruction in said non-branching cluster.

18. (Original) The processing system as set forth in Claim 17 wherein said remote conditional branching control circuitry selects one of said computed next program counter address and said computed branch address in response to said computed branch condition.

19. (Original) The processing system as set forth in Claim 18 wherein said remote conditional branching control circuitry comprises a multiplexor having an input channel associated with said non-branching cluster, said multiplexor responsive to said computed branch condition.

20. (Original) The processing system as set forth in Claim 14 wherein said data processor issues a shadow conditional branch instruction in said branching cluster to perform said branch address computation in response to sensing said conditional branch instruction in said non-branching cluster.

21. (New) A data processor having a clustered architecture that comprises an instruction cache and a plurality of clusters, each cluster comprising an instruction execution pipeline, each instruction execution pipeline comprising a plurality of processing stages, each processing stage capable of performing at least one of a plurality of execution steps associated with instructions being executed by the clusters, the data processor comprising:

a power-down controller capable of monitoring each instruction execution pipeline and the instruction cache to identify one or more power-down conditions associated therewith, wherein the power-down controller is also capable of: (i) bypassing performance of at least a portion of subsequent ones of the processing stages associated with an executing instruction; (ii) powering down the instruction cache; and (iii) powering down the data processor, and wherein the power-down controller, in response to an identified power-down condition, is further capable of at least one of:

(i) bypassing performance of at least the portion of subsequent ones of the processing stages associated with the executing instruction;

(ii) powering down the instruction cache; and

(iii) powering down the data processor.

22. (New) The data processor as set forth in claim 21, further comprising an instruction fetch buffer, and wherein the identified power-down condition indicates detection of at least one of: (i) a non-operation in one of the clusters, (ii) a tight-loop condition in the instruction fetch buffer, and (iii) an idle-loop condition.

23. (New) The data processor as set forth in claim 21, wherein the power-down controller, while monitoring each instruction execution pipeline and the instruction cache, is capable of detecting a non-operation associated with the instruction executing in the instruction execution pipeline of one of the clusters.

24. (New) The data processor as set forth in claim 23, wherein the power-down controller, in response to detecting the non-operation, is capable of bypassing performance of at least the portion of subsequent ones of the processing stages to thereby reduce power consumption in the subsequent ones of the processing stages as the executing instruction passes through the instruction execution pipeline.

25. (New) The data processor as set forth in claim 21, further comprising an instruction-fetch buffer.

26. (New) The data processor as set forth in claim 25, wherein the power-down controller is capable of powering down the instruction cache in response to identifying a tight-loop condition in the instruction fetch buffer.

27. (New) The data processor as set forth in claim 25, wherein the power-down controller is capable of powering down the data processor in response to identifying an idle-loop condition.

28. (New) For use in a data processor having a clustered architecture, the data processor comprising an instruction cache and a plurality of clusters, each cluster comprising an instruction execution pipeline, each instruction execution pipeline comprising a plurality of processing stages, each processing stage capable of performing at least one of a plurality of execution steps associated with instructions being executed by the clusters, the data processor further comprising a power-down controller capable of: (i) bypassing performance of at least a portion of subsequent ones of the processing stages associated with an executing instruction; (ii) powering down the instruction cache; and (iii) powering down the data processor, a method of operating the data processor comprising the steps of:

monitoring each instruction execution pipeline and the instruction cache to identify power-down conditions associated therewith using the power-down controller; and

in response to an identified power-down condition, at least one of: (i) bypassing performance of at least the portion of subsequent ones of the processing stages associated with the executing instruction, (ii) powering down the instruction cache, and (iii) powering down the data processor.

29. (New) The method as set forth in claim 28, further comprising the step of detecting, while monitoring each instruction execution pipeline and the instruction cache, a non-operation associated with the instruction executing in the instruction execution pipeline of one of the clusters.

30. (New) The method as set forth in claim 29, further comprising the step of bypassing, in response to detecting the non-operation, performance of at least the portion of subsequent ones of the processing stages to thereby reduce power consumption in the subsequent ones of the processing stages as the executing instruction passes through the instruction execution pipeline.

31. (New) The method as set forth in claim 29, wherein the detecting step further comprises detecting that the non-operation is one of a real non-operation and an inserted non-operation during a decode stage of the processing stages.

32. (New) The method as set forth in claim 28, wherein the data processor further comprises an instruction fetch buffer, and the method further comprises the step of powering down the instruction cache in response to identifying a tight-loop condition in the instruction fetch buffer.

33. (New) The method as set forth in claim 28, wherein the data processor further comprises an instruction fetch buffer, and the method further comprises the step powering down the data processor in response to identifying an idle-loop condition.

34. (New) A processing system, comprising:

a data processor having a clustered architecture;

a memory associated with the data processor; and

a plurality of peripheral circuits associated with the data processor and capable of performing selected functions in association with the data processor;

wherein the data processor comprises:

an instruction cache;

a plurality of clusters, each cluster comprising an instruction execution pipeline having a plurality of processing stages, each processing stage capable of performing at least one of a plurality of execution steps associated with instructions being executed by the clusters; and

a power-down controller capable of monitoring each instruction execution pipeline and the instruction cache to identify one or more power-down conditions associated therewith, wherein the power-down controller is also capable of: (i) bypassing performance of at least a portion of subsequent ones of the processing stages associated with an executing instruction; (ii) powering down the instruction cache; and (iii) powering down the data processor, and wherein the power-down controller, in response to an identified power-down condition, is further capable of at least one of:

(i) bypassing performance of at least the portion of subsequent ones of the processing stages associated with the executing instruction;

(ii) powering down the instruction cache; and

(iii) powering down the data processor.

35. (New) The processing system as set forth in claim 34, further comprising an instruction fetch buffer, and wherein the identified power-down condition indicates detection of at least one of: (i) a non-operation in one of the clusters, (ii) a tight-loop condition in the instruction fetch buffer, and (iii) an idle-loop condition.

36. (New) The processing system as set forth in claim 34, wherein the power-down controller, while monitoring each instruction execution pipeline and the instruction cache, is capable of detecting a non-operation associated with the instruction executing in the instruction execution pipeline of one of the clusters.

37. (New) The processing system as set forth in claim 36, wherein the power-down controller, in response to detecting the non-operation, is capable of bypassing performance of at least the portion of subsequent ones of the processing stages to thereby reduce power consumption in the subsequent ones of the processing stages as the executing instruction passes through the instruction execution pipeline.

38. (New) The processing system as set forth in claim 34, further comprising an instruction-fetch buffer.

39. (New) The processing system as set forth in claim 38, wherein the power-down controller is capable of powering down the instruction cache in response to identifying a tight-loop condition in the instruction fetch buffer.

40. (New) The processing system as set forth in claim 38, wherein the power-down controller is capable of powering down the data processor in response to identifying an idle-loop condition.